An Evaluation of the TransFER Model for Sharing Clinical Decision-Support Applications

Walter Sujansky, M.D., Ph.D., Russ Altman, M.D., Ph.D.

Section on Medical Informatics
Stanford University School of Medicine, Stanford, CA

TransFER is a formal model designed to facilitate the sharing of decision-support applications across institutions with heterogeneous clinical databases. The TransFER model provides a mechanism to automatically customize database queries based on a reference schema of clinical data and an encoded set of database mappings. In this paper, we describe the elements of the TransFER model and we present the results of a formal evaluation we conducted to assess the utility and generality of the model. The results suggest that the TransFER has significant potential for automating query translation and facilitating application sharing, but that further work on the representation of temporal semantics, on the modeling of missing data, and on the optimization of complex queries is required.

INTRODUCTION

The development of effective Clinical Decision Support (CDS) applications is a costly and time-consuming process that few healthcare institutions can undertake. It is, therefore, desirable to share proven CDS applications among many institutions. One of the major obstacles to sharing CDS applications that access patient data automatically is the heterogeneous nature of existing clinical databases [1]. Database heterogeneity assumes several forms, including data model heterogeneity, structural heterogeneity, naming heterogeneity, and semantic heterogeneity [2]. To share CDS applications across provider sites with heterogeneous databases, one must resolve differences between the representation of clinical data that CDS applications assume and the representations that existing databases provide. For example, to share clinical alert rules encoded in the Arden syntax, local programmers must *customize* the database queries in Arden rules to correspond to the idiosyncratic data models, database schemas, and clinical vocabularies of local information systems [3]. This manual translation of database queries is a laborious, costly, and error-prone process that can prevent effective sharing of CDS applications [1][4].

We have developed an experimental technology, called TransFER, that addresses the impediments to sharing posed by heterogeneous clinical databases. The goal of the TransFER model is to *automatically translate* queries that appear in CDS applications to equivalent queries that are specific to heterogeneous relational databases. We have addressed many of the design, implementation, and optimization issues of the TransFER model previously [4][5]. However, an equally important challenge was to define valid criteria for the evaluation of the TransFER model and to conduct an experiment that assessed TransFER with respect to these criteria. In this paper, we describe the technical content of TransFER briefly, and focus on our evaluation of the TransFER model. Section 2 describes the components of the TransFER model and how they support the sharing of CDS applications. Section 3 proposes a set of criteria by which one can evaluate models designed to facilitate application sharing through automated query translation. In Section 4, we describe the design of the experiment that we conducted to evaluate the TransFER model, and in Section 5 we report and discuss our results.

THE TRANSFER MODEL

TransFER consists of four general components.

1. A **semantic data model** called FER (Functional Entity-Relationship model). FER supports the modeling of clinical data in an abstract fashion that subsumes many site-specific database representations. Figure 1 shows a sample FER schema and enumerates the FER modeling constructs. The FER model is used to specify a global reference schema of clinical data that represents the medical concepts and the associations among concepts typically stored in clinical information systems.

2. A **high-level query language** called ReFER. ReFER corresponds to the FER data model and allows developers of CDS applications (for example, the authors of clinical alert rules) to formulate all data requests as ReFER queries against the FER reference schema. For example, the ReFER query in Figure 2 requests all of the glucose values for the patient

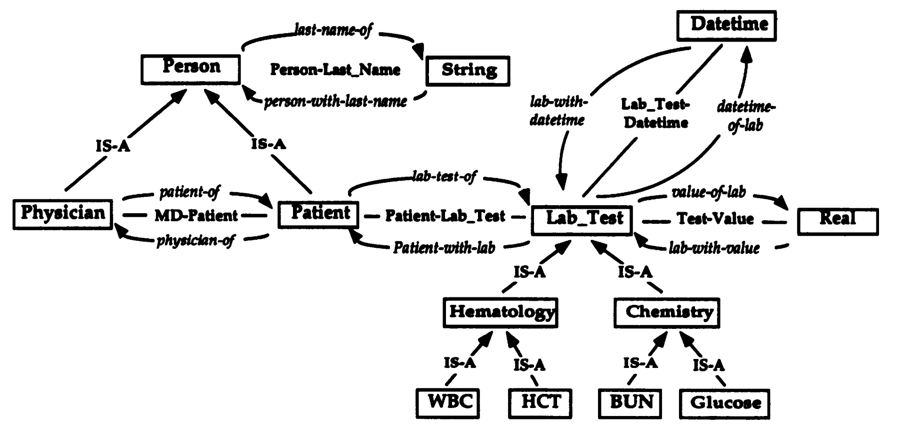

Figure 1. A representative FER schema. Boxes denote entity types; lines annotated with bold-faced text denote relationship types,; arrows annotated with italicized text denote relationship functions; arrows annotated by IS-A labels denote IS-A connections. This schema illustrates a subset of the FER schema used in the evaluation experiment (see Section 5).

named "Smith" between 5:00 and 8:00 AM on October 28th:

```
[ value-of-test(g) | (g:Glucose), (pt:Patient) AND
    Patient-Lab_Test(pt,g) AND
    datetime-of-lab(g) >= "1995/10/28 0500" AND
    datetime-of-lab(g) <= "1995/10/28 0800" AND
    last-name-of(pt) = "Smith" ]
```

Figure 2. A sample query formulated in the ReFER query language with respect to the FER schema in Figure 1.

3. A **mapping language** called ERA (Extended Relational Algebra). ERA allows local database administrators to map each entity type and relationship type of the global FER schema to a relational *view* over the tables and fields of the local database; when correctly specified, the views express the same semantics as the corresponding FER constructs. In certain cases, it may not be possible to define equivalent views using standard relational algebra operators (such as SELECT, JOIN, and UNION). These cases require the specification of *external functions* in ERA expressions; external functions can perform arbitrary operations over relational data, but must be executed outside of the relational database environment. The inclusion of external functions within relational algebra expressions is one of the extensions to the standard relational algebra that we have made.

4. A **query-translation module**. The module automatically translates any ReFER query to an equivalent relational algebra expression by systematically combining the mapping views that correspond to each FER construct in the query. TransFER subsequently optimizes and translates the expression into a site-specific database program, typically consisting of multiple SQL statements and (if necessary) external function calls. When a CDS application is distributed to local provider sites, a TransFER compiler automatically applies the local mappings to translate all ReFER queries that appear in the application into site-specific database programs. These programs access the local database and retrieve the appropriate data when the application is run.

Details of the formal models and query-translation methods used by TransFER appear in [4] and [5].

EVALUATION CRITERIA

To formally evaluate whether TransFER is an effective model for sharing decision-support applications, we specified a set of performance criteria that we could objectively evaluate in the context of a controlled experiment. The five criteria are:

1. **Latitude**: the ability of the ERA mapping language to fully map the FER schema of clinical data to most legacy relational databases, and, therefore, to support automated query translation.

2. **Declarativeness**: the ability of the ERA mapping language to map the FER schema to most legacy relational databases without requiring extensive use of external functions. Because external functions are procedurally specified and encapsulated (as opposed relational algebra operators, which are declarative), they cannot be inspected and manipulated by the TransFER query optimizer, resulting in less efficient translations of queries.

3. **Expressiveness**: The ability of the FER data model and the ReFER query language to represent the semantics of clinical queries that are typically required by decision-support applications.

4. **Correctness**: The ability of the TransFER compiler to generate relational queries that are semantically equivalent to the input ReFER queries.

5. **Efficiency**: the ability of the TransFER query optimizer to generate relational queries that are

Table 1. Success rates for mapping the FER schema to relational schemas

Schema	Mapping Yield (%)	External Function Use (%)
Schema 1	145/183 (79%)	12/145 (8%)
Schema 2	183/183 (100%)	13/183 (7%)
Schema 3	183/183 (100%)	13/183 (7%)

efficient enough for clinical decision-support applications. Poor efficiency would indicate that manual query translation is still required to provide sufficiently fast database access for clinical applications.

THE EVALUATION EXPERIMENT

Ultimately, the performance criteria must be applied to TransFER in the context of sharing real CDS applications among actual provider sites. However, at this stage in development, we chose to perform a more modest controlled experiment to evaluate whether TransFER could meet a representative subset of the challenges it will encounter in the real world. The experiment entailed six steps:

1. Three volunteer subject groups, each consisting of one clinician and one database expert, designed and populated a relational database schema based on the same clinical data set.

2. With no knowledge of the relational schemas under construction, we designed a global reference schema in the FER conceptual-modeling language to represent the same clinical data set. An excerpt of this schema appears in Figure 1.

3. Two physicians formulated a set of 23 natural-language queries to retrieve clinically relevant information that appeared in the data set. The queries ranged in complexity from "What are the allergies of Patient 1?" to "How many hours after Patient 2 is placed on Cefotetan does her temperature fall below 101?" We subsequently encoded each of these queries in the ReFER query language such that they were consistent with the FER reference schema.

4. We mapped the FER reference schema to each relational schema using the ERA mapping language.

5. The TransFER compiler translated the ReFER queries to three sets of SQL queries such that each set was consistent with one of the relational database schemas.

6. We manually translated the ReFER queries to three sets of "gold-standard" SQL queries, which we compared to the TransFER-generated queries with respect to the correctness and efficiency criteria.

RESULTS AND DISCUSSION

The evaluation experiment yielded useful data for assessing the effectiveness of the TransFER technology with respect to each performance criterion. Tables 1 – 3 present the results of the experiment.

Latitude and Declarativeness

Table 1 addresses the performance of TransFER with respect to the latitude and declarativeness criteria. The "Mapping Yield" column indicates the proportion of entity types and relationship types in the FER reference schema that we successfully mapped to each of three relational schemas using the ERA mapping language. Although schemas 2 and 3 were entirely mapped to the FER reference schema, schema 1 proved problematic because its designers chose not to model clinical diagnoses and medical procedures in a structured relational format. In schema 1, all of the diagnoses and procedures associated with each patient were represented in a *single text field* that contained a narrative description of the patient's chief complaint and past medical history. The extraction of discrete diagnosis and procedure entities from the free-text narrative was beyond the capabilities of the relational algebra or any external functions available to us. In this sense, schema 1 was beyond the "latitude" of the TransFER database-mapping model. However, in the absence of natural-language processing techniques [6] (which are as yet immature), the diagnosis and procedure data in schema 1 are beyond the latitude of any query-translation model, including manual query translation. In this sense, the evaluation shows that the TransFER model has latitude *at least as* great as any current method for sharing decision-support applications.

The "External Function Use" column of Table 1 denotes the proportion of ERA mapping views that required at least one external function call to accurately represent the semantics of the corresponding FER entity or relationship type. As the data indicate, fewer than 10 percent of the mapping views required the use of an external function. Given the degree of heterogeneity among the generated relational schemas, this result reflects the power of the relational algebra as a mapping language and foreshadows the potential for significant optimization of automatically translated queries (as was borne out by the efficiency data).

Table 2. Rates of success for encoding, compiling, and executing queries

Parameter	Value
% of physician queries encoded in ReFER	70% (16/23)
% of translated queries returning correct values	92% (36/39)

Table 3. Parameters characterizing the complexity of generated queries

Parameter	Manual translation	TransFER translation (no optimization)	TransFER translation (partial optimization)
Average # SQL queries/ReFER query	1.1 (1–2)	5.5 (3–11)	3.5 (1–9)
Average JE/ReFER query	4.4 (2–9)	22.9 (9–54)	17.1 (2–44)

Expressiveness and Correctness

Table 2 addresses the performance of TransFER with respect to the expressiveness and correctness criteria. The ability of the ReFER query language to express the clinical queries formulated in step 3 of the experiment reflects the utility of the TransFER query-translation model for sharing *clinical* decision-support applications. The data in Table 2 shows that 16 of the 23 queries specified by physicians as relevant to decision support (70%) could be encoded in the ReFER query language. Of the remaining seven queries, five could not be encoded because ReFER currently lacks constructs to perform certain temporal operations, such as retrieving the three most recent values of a laboratory test or computing the difference in hours between two time points. These data imply that the TransFER model would benefit from more powerful temporal operators. However, in the absence of such operators, the TransFER model still provides an effective means of sharing CDS applications because the applications can use the ReFER language to retrieve a superset of the requested data from the database and subsequently can perform the temporal operations themselves.

Each of the 16 successfully encoded ReFER queries subsequently was processed by the translating compiler three times, once using each set of schema-specific ERA mappings. Of the 48 potential query translations, 39 produced valid database programs. The remaining nine translations produced invalid programs because at least one of the generated SQL queries referenced more than 16 relational tables, an arbitrary limit of our database software (Sybase). The result of each successful translation was a database program consisting of SQL queries interspersed with external function calls. When executed against the appropriate relational database, the program ostensibly returned the data requested in the original ReFER query. Any discrepancy between the data originally requested and the data ultimately retrieved violated the correctness criterion.

36 of the 39 database programs (92%) produced results that were correct. One of the three incorrect queries resulted from an erroneous ERA mapping expression; when we corrected the mapping error and retranslated the query, the appropriate data was retrieved. Mapping errors may be subtle and remain undetected in the absence of rigorous testing, which suggests the need for extensive test suites of standardized data and queries to help detect errors. Another query was incorrect because one of subject groups populated their relational schema in a manner inconsistent with the semantics of the original clinical data set; after we modified the data, the same query produced correct results. The third incorrect query occurred because one relational schema contained no data regarding patient allergies, although this data was included in the clinical data set. When we mapped the FER relationship type denoting each patient's allergies to this schema, we assumed that patients had no allergies (in the absence of data to the contrary), which produced incorrect query results. Missing clinical data elements pose a significant problem for the sharing of CDS applications, because incorrect assumptions regarding the values of such elements may affect the validity of inferences in ways that are not apparent to users. One method of addressing this problem is to augment the TransFER model such that mappings can assign a special value to missing data elements that notifies the decision-support application that requested data is not available. This strategy would allow an application to incorporate into its logic the possibility that certain data are missing and to respond in an application-specific and appropriate manner (for example, by querying the user).

Efficiency

Table 3 addresses the efficiency criterion. To assess this criterion, we directly compared the 39 TransFER-generated database programs to a corresponding set of *manually* translated programs for the same ReFER queries. The manually translated database programs constituted a *gold standard* of efficiency in the sense that we formulated these queries with the intent of specifying the minimal set of SQL operations required to retrieve the requested data. We characterize the complexity of each database program generated by TransFER in terms of two parameters: (1) the number of distinct SQL queries in each database program and (2) the number of *Join Equivalents* in each database program. We define the number of Join Equivalents (JEs) in a database program as the total number of joins across all of the SQL queries in the program *plus* two times the number of SQL queries in the program. The assumption underlying this metric is that the overhead associated with each distinct SQL query in a database program is roughly equivalent to two relational joins. This definition implies, for example, that a database program consisting of five SQL queries that each contain one join (JE = 15) is twice as costly to execute as a database program consisting of one SQL query containing five joins (JE = 7). We use relational joins as a gross measure of complexity because they typically constitute the bulk of the run-time processing of SQL queries.

Table 3 shows the average number of SQL queries ("Average SQL queries/ReFER query") and the average number of join equivalents ("Average JE/ReFER query") for the entire set of 39 database programs generated by the TransFER compiler. The parameters were calculated once before any compiler optimizations were implemented (column 3 of Table 3) and once after a subset of the optimizations that we devised was implemented (column 4 of Table 3). Column 2 of Table 3 shows the average number of SQL queries and the average number of join equivalents for the gold-standard database programs that we generated manually. It is useful to note that the 39 ReFER queries that were translated contained, on average, one term, two declarations, and four predicates (similar to the sample query in Figure 2). The data in Table 3 indicate that the automatically generated database programs (before optimization) are more complex than the ideal (gold-standard) programs by a factor of five to one. Direct comparison of corresponding database programs confirms that superfluous JOIN and UNION operations in the TransFER-generated programs are responsible for this disparity in complexity. However, the data also indicate that implementation of just a subset of the possible optimizations, which eliminates certain redundant JOIN and UNION operations, results in a 20 percent reduction in complexity. This result is promising because it indicates that optimization techniques based on the principles of relational algebra can significantly improve the efficiency of TransFER-generated queries. The development and implementation of additional optimization rules will likely lead to further performance improvements, bringing the efficiency of TransFER-generated database programs closer to that of the gold standard.

CONCLUSIONS

Our formal evaluation of the TransFER model showed that (1) the FER data model and ReFER query language are sufficiently expressive to represent most queries that appear in CDS applications, (2) the ERA mapping language is sufficiently powerful to encode mappings between FER schemas and structured relational databases, and (3) the TransFER compiler is sufficiently robust to translate ReFER queries into equivalent SQL queries that are site-specific, provided that the ERA mappings are correct and that local databases are complete with respect to the FER schema. The evaluation also identified several inadequacies of the model, which has directed us towards adding temporal operators to the ReFER query language, augmenting the ERA mapping language with constructs to explicitly denote missing data, and expanding the optimization techniques available to the TransFER compiler. Given the demonstrated soundness of the basic query-translation model, these extensions may well complete the link provided by TransFER between decision-support applications and heterogeneous relational databases.

Acknowledgments: Portions of this work were supported by NLM contracts N01-LM-4-3514 and LM-05652, the CAMIS resource under NLM grant LM05305, and Culpeper Medical Scholars.

References

1. Pryor, T. A.& Hripcsak, G. (1994). Sharing MLMs: An experiment between Columbia-Presbyterian and LDS Hospital. In C. Safran (Ed.), *Proceedings of the Symposium on Computer Applications in Medical Care*. New York: McGraw-Hill, pp. 399-403.
2. Deen, S.M., et. al. (1987). Data integration in distributed databases. *IEEE Transactions on Software Engineering*. Vol. SE-13, No. 7, pp. 860-864.
3. Hripcsak, G., et. al. (1990). The Arden Syntax for Medical Logic Modules. In R. Miller (Ed.), *Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care*. Washington, D.C.: , pp. 200-204.
4. Sujansky, W. and Altman, R. (1994). Towards a standard query model for sharing decision-support applications. In J. Ozbolt (Ed.), *Proceedings of the Symposium on Computer Applications in Medical Care*. Philadelphia: Hanley & Belfus, pp. 325-331.
5. Sujansky, W. (1996). *A Formal Model for Bridging Heterogeneous Relational Databases in Clinical Medicine*. Ph.D. Thesis. Section on Medical Informatics, Stanford University.
6. Sager, N., et. al. (1994). Natural language processing and the representation of clinical data. *Journal of the American Medical Informatics Association*, 1(2): 142-160.